Projects...

- Project #1:
  - Implement simple distance vector routing protocol
  - We provide a “framework” called fisnhet for implementing
    » Fishead: program that simulates network; maintains topology, etc.
    » Libfish.a: library that provides basic functions (sending and receiving packets, timers and keyboard input)
    » Fish.h: a header file that defines this API
  - You will test your implementation in your own local fishnet
  - We will provide a long-running fishnet that everyone in the class can join (one big network)

  BUT… still not ready… 😊 Tuesday with complete documentation. Will be due two weeks from Tuesday.

Last class

- Routing: how to get packets to their destination
  - Forwarding: local calculation to decide next hop for each packet
  - Routing: global calculation to ensure that forwarding decisions ultimately take packets to the right place

Intra-domain routing protocols
  - Also called Interior Gateway Protocols (IGP)
  - Distance Vector
    » Local exchange of global routing information
    » In steady-state converges to correct solution
    » Problems during failures: count-to-infinity

This class

- Finish Intra-domain routing
  - Link-state protocols

Inter-domain routing
  - BGP
  - Policy
  - Peering/transit economics

Link State routing

- Same goal as DV, but a different approach
- Two phases
  - Reliable flooding
    » Tell all routers what you know about your local topology
  - Path calculation (Dijkstra’s algorithm)
    » Each router computes best path over complete network

Motivation
  - Using DV, routers only have local information, making it difficult to decide what to do when there are changes
  - With LS, faster convergence and better stability (hopefully)
  - More complex

Flooding

- Each router maintains link state database and periodically sends link state packets (LSPs) to neighbor
  - LSPs contain [router, neighbors, costs]
- Each router forwards LSPs not already in its database on all ports except where received
  - Each LSP will travel over the same link at most once in each direction
- Flooding is fast, and can be made reliable with acknowledgments
Reliable flooding

- Goal: tell everyone what you know about local topology
- Periodically send link state packets (LSPs) on all links
  - LSP contains [node, neighbors, costs, sequence number]
- If node X receives an LSP from node Y over link Q
  - Save it in local link state database
  - Forward LSP on all links except Q
- Use explicit ACKs and retransmits to make flooding reliable
- Each LSP will travel at most once over each link

Flooding example

- LSP generated by X at T=0
- Nodes become orange as they receive it

More challenges

- What happens if the network is partitioned and heals?
  - Different LS databases must be synchronized
  - Use version #s on each LSP (incremented for each update)
  - Compare version #s when a link comes back up and request out of date LSPs

Dijkstra’s Shortest Path Tree (SPT) algorithm

- Graph algorithm for single-source shortest path tree

```plaintext
S ← {};
Q ← <all nodes keyed by distance>
While Q ≠ {}:
  u ← extract-min(Q)
  S ← S plus {u}
  for each node v adjacent to u
  "relax" the cost of v
```

Dijkstra Example – Step 1
Example – Step 2

Example – Step 3

Example – Step 4

Example – Step 5

Example – Done

Link State evaluation

- Strengths
  - Loop free as long as LSDB’s are consistent
    - Can have transient routing loops
  - Messages are small (esp compared to DV)
  - Converges quickly (esp compared to DV)

- Weaknesses
  - Must flood data across entire network (scalability?)
  - Must maintain state for entire topology
Link State in practice

- OSPF (Open Shortest Path First) and IS-IS
  - Most widely used intra-domain routing protocol
  - Run by almost all ISPs and many large organizations
- Basic link state algorithm plus many features:
  - Authentication of routing messages
  - Extra hierarchy: Partition into routing areas
  - Load balancing: Multiple equal cost routes

Historic context

- Original ARPAnet had single routing protocol
  - Dynamic DV scheme, replaced with static metric LS algorithm
- New networks came on the scene
  - NSFnet, CSnet, DDN, etc...
  - The total number of nodes was growing exponentially
  - With their own routing protocols (RIP, Hello, ISIS)
  - And their own rules (e.g. NSF AUP)
- Scalability: Routing tables with millions of entries?
- Heterogeneity: Network A uses hop count as a metric, Network B uses measured delay, Network C uses link capacity; what if networks use different routing protocols?
- Policy: Network A connects to Networks B and C. Network B is only allowed to carry network C’s traffic?

But the Internet is not just one network...

- Inter-domain versus intra-domain routing

Solution: Inter-domain routing

- Separate routing inside a domain from routing between domains
  - Inside a domain use traditional interior gateway protocols (RIP, OSPF, etc)
  - Between domains use Exterior Gateway Protocols (EGPs)
    - Only exchange reachability information (no metrics)
    - Decide what to do based on local policy
- Terminology: Autonomous Systems (ASes)
  - Unit of abstraction in interdomain routing; another word for domain
  - Roughly, a network with common administrative control, a coherent internal routing policy, and presenting a consistent external view of connectivity
  - Represented by a 16-bit number
    - Example: UUnet (701), Sprint (1239), UCSD (7377)

Inter-Domain Routing

- Network comprised of many Autonomous Systems (ASes) or domains
- To scale, use hierarchy: separate inter-domain and intra-domain routing
- Also called interior vs exterior gateway protocols (IGP/EGP)
  - IGP = RIP, OSPF
  - EGP = EGP, BGP

Inter-Domain Routing

- Border routers summarize and advertise internal routes to external neighbors and vice-versa
- Border routers apply policy
- Internal routers can use notion of default routes
- Core is "default-free"; routers must have a route to all networks in the world
**Exterior Gateway Protocol**

- First major inter-domain routing protocol
- Spanning tree: no loops

**Problems with EGP**
- In 1995 NSFnet got out of the backbone business
  - Many backbones (MCI, Sprint, AT&T…)
  - Multiconnected regional networks
  - Meshed topology, loops…
- A tree-based structure didn’t work anymore
- Need a new protocol…

**What kind of protocol?**

- Link state?
  - Too much state
    - Currently 11,000 ASs and > 100,000 networks
  - Relies on global metric & policy
- Distance vector?
  - May not converge; loops
  - Relies on global metric and policy
- Solution: path vector
  - Reachability protocol, no metrics
  - Route selection based on local policy
  - Route advertisements carry list of ASs
    - "I can reach UCSD through this path: AS73, AS703, AS1"
  - Automatic loop detection. Why? How?

**Path Vectors**

- Similar to distance vector, except send entire paths
  - e.g. 321 hears [7,12,44]
  - stronger avoidance of loops
  - supports policies (later)
- Modulo policy, shorter paths are chosen in preference to longer ones
- Reachability only – no metrics

**Policies**

- Choice of routes may depend on owner, cost, AUP, …
  - Business considerations (more on this later)
- Local policy dictates what route will be chosen and what routes will be advertised!
  - e.g., X doesn’t provide transit for B, or A prefers not to use X

**How BGP operates (roughly)**

- Establish session on TCP port 179
- Exchange all active routes
- Exchange incremental updates
- Pros/Cons of using TCP?
### Two types of BGP neighbor relationships

- External Neighbor (eBGP) in a different Autonomous Systems
- Internal Neighbor (iBGP) in the same Autonomous System

### Why do we need iBGP?

- iBGP neighbors do not announce routes received via iBGP to other iBGP neighbors.
- iBGP is needed to avoid routing loops within an AS.
- Injecting external routes into IGP does not scale and causes BGP policy information to be lost.

### Important BGP attributes

- **Local pref**: Statically configured ranking of routes within AS
- **AS path**: ASs the announcement traversed
- **Origin**: Route came from IGP or EGP
- **Multi Exit Discriminator**: preference for where to exit
- **Community**: opaque data used for inter-ISP policy
- **Next-hop**: where the route was heard from

### BGP Decision process

- Default decision for route selection
  - Highest local pref, shortest AS path, lowest MED, prefer eBGP over iBGP, lowest IGP cost, router id
- Many policies built on default decision process, but...
  - Possible to create arbitrary policies
  - Can have separate policy for inbound routes, installed routes and outbound routes
  - Limited only by power of vendor-specific routing language

### Example: local pref

Higher Local preference values are more preferred

### Example: AS Path

Shorter AS Paths are more preferred
Mr. BGP says that path 4 > 1 is better than path 3 2 1. The shortest AS path doesn’t mean the best path.

This Router has two BGP routes to 192.44.78.0/24. Hot potato: get traffic off of your network as soon as possible. Go for egress 1!

Problems with hot potato:
- High bandwidth Provider backbone
- Low bandwidth customer backbone
- Many customers want their provider to carry the bits!

Ongoing Problems w/BGP:
- Instability
  - Route flapping
  - Long AS-path decision criteria defaults to DV-like behavior (bouncing)
  - Not guaranteed to converge, NP-hard to tell if it does
- Scalability still a problem
  - ~100,000 network prefixes in default-free table today
  - Tension: Want to manage traffic to very specific networks (e.g., multihomed content providers) but also want to aggregate information.
- Performance
  - Non-optimal, doesn’t balance load across paths
- Security…

Routing policy:
- So far we’ve discussed mechanism…
- How and why are basic routing policies decided?

History:
- First policies for political reasons
  - NSFnet AUP (even today Internet2)
- Emergence of commercial policies
  - 1994-1995 NSFnet transition
    - NSF ceases to run Internet backbone
    - Commercial carriers (MCI, Sprint, ANS) start selling IP backbone service
    - Interconnected with each other and regional networks at several public NAPs
    - Everyone talks to everyone
  - Then five years went by…
Background – Settlement

- The telephone world
  - LECs (local exchange carriers)
  - IXCs (inter-exchange carriers)
- LECs MUST provide IXCs access to customers; regulation
- When a call goes from one phone company to another:
  - Call billed to the caller
  - The money is split up among the phone systems – this is called “settlement”

On the Internet...

- No regulation
  - One ISP doesn’t have to talk to another
- Founded on “shared goodwill”
  - Pay for connectivity, not per packet
  - Not clear who should pay anyway
  - No standard settlement

Peering vs Transit

- Peering
  - Two ISPs provide connectivity to each others customers (traditionally for free)
  - Non-transitive relationship
- Transit
  - One ISP provides connectivity to every place it knows about (usually for money)

Example: peering

Example: transit

By EastNet purchasing transit, Eastnet is announced by USNet to USNet peering and transit interconnections alike.

Example: transit (2)

The entire Internet as known by USNet
**The value of transit**

- Not just paying for the fiber, but the connectivity
  - Remember, there is no single “backbone”
  - If you’re an ISP, how do your customers get to yahoo.com?
- Means big ISPs have more value to offer small ISPs than vice-versa

**Aside...**

- Peering and transit are really two popular points on a continuum
- Some places sell “partial transit”
- Other places sell “usage-based” peering
- Principle issue is:
  - Which routes do you give away and which do you sell? To whom? Under what conditions?

**Summary**

- Link-state intra-domain routing
  - Tell everyone about your neighbors
  - Low message overhead, good convergence
  - Must maintain lots of state
- Interdomain-routing
  - Exchange reachability information (plus hints)
  - Local policy to decide which path to follow
- Traffic exchange policies are a big issue $$$
  - Complicated by lack of compelling economic model (who creates value?)
  - Can have significant impact on performance

**For next time...**

- Mobile and Multicast routing...
- Chapter 4.2.5 and 4.4